**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

# Лабораторная работа № 20

**Студент:** Керимов А. Ш.

**Группа:** ИУ7-64Б

**Преподаватель:** Толпинская Н. Б.

Москва.
2020 г.

**Цель работы** – изучить способы формирования и модификации списков в Prolog, эффективные методы обработки списков и порядок реализации рекурсивных программ.

**Задание.**

**Ответить на вопросы (**коротко**):**

1. Как организуется хвостовая рекурсия в Prolog?
2. Какое первое состояние резольвенты?
3. Каким способом можно разделить список на части, какие, требования к частям?
4. Как выделить за один шаг первые два подряд идущих элемента списка? Как выделить 1-й и 3-й элемент за один шаг?
5. Как формируется новое состояние резольвенты?
6. Когда останавливается работа системы? Как это определяется на формальном уровне?

**Используя хвостовую рекурсию, разработать, комментируя аргументы, эффективную программу, позволяющую:**

1. Сформировать список из элементов числового списка, больших заданного значения;
2. Сформировать список из элементов, стоящих на нечётных позициях исходного списка (нумерация от 0);
3. Удалить заданный элемент из списка (один или все вхождения);
4. Преобразовать список в множество (можно использовать ранее разработанные процедуры).

   Убедиться в правильности результатов

**Для одного** из вариантов **ВОПРОСА** и **1-ого задания составить таблицу**, отражающую конкретный порядок работы системы:

Т.к. резольвента хранится в виде стека, то состояние резольвенты требуется отображать в столбик: вершина – сверху! Новый шаг надо начинать с нового состояния резольвенты! Для каждого запуска алгоритма унификации, требуется указать № выбранного правила и соответствующий вывод: успех или нет –и почему.

**Текст процедуры …; Вопрос:…..**

| № шага | Текущая резольвента – ТР | ТЦ, выбираемые правила: сравниваемые термы, подстановка | Дальнейшие действия с комментариями |
|---|---|---|---|
| шаг1 | … | … | … |
| … | … | … | … |

# Практическая часть

*Листинг 1. Факториал и Фибоначчи*

```
domains
  elements = integer*

predicates
  greater(elements List, integer Num, elements Res)
  oddList(elements List, elements Res)
  delFirst(elements List, integer Num, elements Res)
  del(elements List, integer Num, elements Res)
  member(integer X, elements List)
  makeSet(elements List, elements Set)

clauses
  % 1. Список чисел больших Num
  greater([], _, []) :- !.
  greater([H|T], Num, [H|T2]) :- H > Num, !, greater(T, Num, T2).
  greater([_|T], Num, Res) :- greater(T, Num, Res).

  % 2. Список чисел на нечетных позициях
  oddList([], []) :- !.
  oddList([_], []) :- !.
  oddList([_, N|T], [N|T2]) :- oddList(T, T2).

  % 3.1. Удаление первого вхождения числа в список
  delFirst([], _, []) :- !.
  delFirst([X|T], X, T) :- !.
  delFirst([Y|T], X, [Y|T2]) :- delFirst(T, X, T2).

  % 3.2. Удаление всех вхождений числа в список
  del([], _, []) :- !.
  del([H|T], Num, [H|T2]) :- H <> Num, !, del(T, Num, T2).
  del([_|T], Num, Res) :- del(T, Num, Res).

  % 4. Преобразовать список в множество
  member(X, [X|_]) :- !.
  member(X, [_|T]) :- member(X, T).

  makeSet([], []) :- !.
  makeSet([H|T], T2) :- member(H, T), !, makeSet(T, T2).
  makeSet([H|T], [H|T2]) :- makeSet(T, T2).

goal
  greater([], 3, Res).                      % Res = []
  %greater([1, 2, 3], 3, Res).              % Res = []
  %greater([1, 2, 3], 0, Res).              % Res = [1, 2, 3]
  %greater([1, 2, 3, 4, 5], 3, Res).        % Res = [4, 5]
  %oddList([], Res).                        % Res = []
  %oddList([1, 2, 3, 4, 5], Res).           % Res = [2, 4]
  %delFirst([], 3, Res).                    % Res = []
  %delFirst([1, 2, 3, 4, 5], 9, Res).       % Res = [1, 2, 3, 4, 5]
  %delFirst([1, 2, 3, 4, 3, 3, 5], 3, Res). % Res = [1, 2, 4, 3, 3, 5
  %del([], 3, Res).                         % Res = []
  %del([1, 2, 3, 4, 5], 9, Res).            % Res = [1, 2, 3, 4, 5]
  %del([1, 2, 3, 4, 3, 3, 5], 3, Res).      % Res = [1, 2, 4, 5]
  %makeSet([], Set).                        % Set = []
  %makeSet([1, 2, 3, 4, 5], Set).           % Set = [1, 2, 3, 4, 5]
  %makeSet([1, 1, 1, 2, 3], Set).           % Set = [1, 2, 3]
```

## Таблица для цели greater([1, 2, 3, 4, 5], 3, Res).

| № шага | Текущая резольвента – ТР | ТЦ, выбираемые правила: сравниваемые термы, подстановка | Дальнейшие действия с комментариями |
|---|---|---|---|
| 1 | greater([1,2,3,4], 2, Res) | ТЦ: greater([1,2,3,4], 2, Res).<br>Поиск с начала базы знаний.<br>П1: greater([], _, []).<br>Сравниваемые термы:<br>[1,2,3,4] = [].<br>Неудача при сопоставлении списков.<br>Возврат к ТЦ, метка переносится ниже.<br><br>ТЦ: greater([1,2,3,4], 2, Res).<br>П2: greater([H\|T], Num, [H\|T2]).<br>Сравниваемые термы:<br>[1,2,3,4]=[H\|T].<br>Успех, подстановка {H=1, T=[2,3,4], Num=2, Res=[1\|T2]}. | Проверка тела П2.<br>Изменение резольвенты в 2 этапа. |
| 2 | 1 > 2<br>!<br>greater([2,3,4], 2, T2) | ТЦ: 1 > 2<br>Неудача. | Откат к предыдущему состоянию резольвенты, метка переносится ниже. |
| 3 | greater([1,2,3,4], 2, Res) | ТЦ: greater([1,2,3,4], 2, Res).<br>П3: greater([_\|T], Num, Res).<br>Сравниваемые термы:<br>[1,2,3,4]=[_\|T].<br>Успех, подстановка {T=[2,3,4], Num=2, Res=Res}. | Проверка тела П3.<br>Изменение резольвенты в 2 этапа. |
| 4 | greater([2,3,4], 2, Res) | ТЦ: greater([2,3,4], 2, Res).<br>Поиск с начала базы знаний.<br>П1: greater([], _, []).<br>Сравниваемые термы:<br>[2,3,4] = [].<br>Неудача при сопоставлении списков.<br>Возврат к ТЦ, метка переносится ниже.<br><br>ТЦ: greater([2,3,4], 2, Res).<br>П2: greater([H\|T], Num, [H\|T2]).<br>Сравниваемые термы:<br>[2,3,4]=[H\|T].<br>Успех, подстановка {H=2, T=[3,4], Num=2, Res=[2\|T2]}. | Проверка тела П2.<br>Изменение резольвенты в 2 этапа. |
| 5 | 2 > 2<br>!<br>greater([3,4], 2, T2) | ТЦ: 2 > 2<br>Неудача. | Откат к предыдущему состоянию резольвенты, метка переносится ниже. |
| 6 | greater([2,3,4], 2, Res) | ТЦ: greater([2,3,4], 2, Res).<br>П3: greater([_\|T], Num, Res).<br>Сравниваемые термы:<br>[2,3,4]=[_\|T].<br>Успех, подстановка {T=[3,4], Num=2, Res=Res}. | Проверка тела П3.<br>Изменение резольвенты в 2 этапа. |
| 7 | greater([3,4], 2, Res) | ТЦ: greater([3,4], 2, Res).<br>Поиск с начала базы знаний.<br>П1: greater([], _, []).<br>Сравниваемые термы:<br>[3,4] = [].<br>Неудача при сопоставлении списков.<br>Возврат к ТЦ, метка переносится ниже.<br><br>ТЦ: greater([3,4], 2, Res).<br>П2: greater([H\|T], Num, [H\|T2]).<br>Сравниваемые термы: | Проверка тела П2.<br>Изменение резольвенты в 2 этапа. |

| | | | |
|---|---|---|---|
| | | [3,4]=[H\|T].<br>Успех, подстановка {H=3, T=[4], Num=2, Res=[3\|T2]}. | |
| 8 | 3 > 2<br>!<br>greater([4], 2, T2) | ТЦ: 3 > 2<br>Успех. | Изменение резольвенты в 2 этапа. |

# Теоретическая часть

**1. Как организуется хвостовая рекурсия в Prolog?**

Организация хвостовой рекурсии:

- Рекурсивный вызов единственен и расположен в конце тела правила.
- До вычисления рекурсивного вызова не должно быть возможности сделать откат (т. е. точки отката отсутствуют). Этого можно добиться, например, с помощью предиката отсечения.

**2. Какое первое состояние резольвенты?**

Если задан простой вопрос, то сначала он попадает в резольвенту.
Если вопрос представляет собой конъюнкцию нескольких термов, то резольвента будет содержать все эти термы, имея на вершине первый терм.

**3. Каким способом можно разделить список на части, какие, требования к частям?**

В Prolog существует более общий способ доступа к элементам списка. Для этого используется метод разбиения списка на начало и остаток. Начало списка – это группа первых элементов, не менее одного. Остаток списка – обязательно список (может быть пустой). Для разделения списка на начало, и остаток используется вертикальная черта (|) за последним элементом начала.
Остаток — это всегда один (простой или составной) терм.

**4. Как выделить за один шаг первые два подряд идущих элемента списка? Как выделить 1-й и 3-й элемент за один шаг?**

Первые два подряд идущих элемента списка:
[H, N2 | _]
где
H – 1-й элемент списка,
N2 – 2-й элемент списка.

1-й и 3-й элемент списка:
[H, _, N3|_]
где
H – 1-й элемент списка,
N3 – 3-й элемент списка.

**5. Как формируется новое состояние резольвенты?**

Изменение резольвенты происходит в 2 этапа:
1) из стека выбирается подцель (верхняя, т.к. стек) и для неё выполняется редукция, т.е. замена подцели на тело найденного правила;
2) к полученной конъюнкции целей применяется подстановка (наибольший общий унификатор выбранной цели и заголовка сопоставленного с этой целью правила).

**6. Когда останавливается работа системы? Как это определяется на формальном уровне?**

Если достигнут конец базы знаний и нет альтернативных путей сопоставления (все метки выбранных ранее правил достигли конца БЗ), то работа завершается.